Formal Semantics for Reactive GRAFCET

Franck Cassez

Formal Semantics for Reactive GRAFCET

Franck Cassez

Département d'Informatique
Université de Bretagne Occidentale
6, Avenue Le Gorgeu B.P. 809
29285 Brest Cedex
FRANCE
e-mail: Franck.Cassez@univ-brest.fr

ABSTRACT. *GRAFCET is a graphical formalism derived from Petri Nets and widely used to program automation applications. So far, this formalism has not been equipped with a formal semantics: interpretation algorithms give the meaning of a GRAFCET description. Our purpose is to take advantage of the work carried out for reactive languages: these languages are given a precise behavioural semantics by means of finite-state machines; the behavioural model can then be checked for various properties. The work presented hereafter consists in equipping GRAFCET with a formal semantics to obtain a behavioural model (namely a timed automaton) that captures the metric aspect of time.*

RÉSUMÉ. *Nous proposons dans cet article une définition du GRAFCET comme langage réactif :* ^{R}GRAFCET. *On définit ensuite une sémantique opérationnelle des programmes de* ^{R}GRAFCET *sous forme de règles à la Plotkin. Cette sémantique inclut les temporisations qui sont traitées comme des variables booléennes de la même manière que les flots d'entrée. On obtient ainsi un modèle de comportement des programmes de* ^{R}GRAFCET *qui est un automate temporisé.*

KEYWORDS: *semantics, GRAFCET, timed automaton, reactive languages*
MOTS-CLÉS: *sémantique, GRAFCET, automate temporisé, langages réactifs*

1. Introduction

The GRAFCET formalism [AFC 77] (Function Charts for Control Systems) is one of the first *language* aimed at specifying real-time applications. It is derived from Petri nets [BRA 83] from which it inherites most of its features: *places* are called *steps*, they are linked together by transitions. We adopt hereafter the terms used for transitions systems (or FSM) and a transition refers to a triple $(step, receptivity, step)$ (in GRAFCET standards the term transition is used for receptivity which is the transition

condition). This formalism is widely used by automation manufacturers and thus has been given a lot of attention regarding standards [COM 88] [AFN 82] [COM 93] for example. Nevertheless the crucial point of the *semantics* given to a GRAFCET description (*a grafcet* in the sequel) has barely been tackled: many *interpretations* have been proposed for GRAFCET [AA 92] and many are used (this implies that GRAFCET programs are bound to be run on particular sites with a definite interpretation and may not be interpreted "equivalently" on different sites).

Recently, this aspect of GRAFCET has benefited from the development of *reactive languages* [PNU 86] [BB 91] [ER 85] [BD 91] [HCRP 91] [LLGL 91] [HAR 87]: a way to give GRAFCET a semantics is to translate GRAFCET programs to reactive programs [RR 94] [AP 92] [ML 92] (if the translation is sufficiently precise) since their semantics is formally defined [BG 91] [CPHP 87] [LBBG 86] [CR 95]. This technique has many advantages as it embeddes GRAFCET into the reactive framework: simulation tools, verification tools are then available for GRAFCET programs. This is very important as GRAFCET specifications can now be verified automatically and connected to reactive languages.

Nevertheless doing so has some drawbacks: the reactive behavioural model is based on *logical* time and only the order of the occurrences of events matters. One of the important features of GRAFCET is the *time-condition* allowing the use of delays in the transition conditions. This notion can not be handled easily with reactive languages. Moreover GRAFCET is a rather complex formalism with many entities like *actions* associated to steps or boolean conditions with integer variables associated with transitions.

The need for a semantic model that handles such features or can be extended to cope with them is quite obvious: it enables to model all the components of the GRAFCET language and to extend the scope of the verification to quantitative aspects.

To give GRAFCET a rigourous and unambiguous interpretation, it remains to fill the gap between the language and a precise behavioural model: this is the purpose of the semantics presented in this paper.

It is not THE semantics of GRAFCET but an attempt to show the possibility to treat this formalism the same way as reactive languages and to benefit from the same advantages: formalization, determinism, verification.

In the next section we introduce the intuitive notions of reactive GRAFCET and justify the choices on particular points of GRAFCET interpretation. Section 3 is devoted to the semantics rules (SOS) and the formal characterization of our GRAFCET interpretation. In section 4 we define the behavioural timed model for GRAFCET specifications.

2. GRAFCET as a reactive language

A *reactive system* [PNU 86] [MP 93] [BB 91] is characterized by the following

features (among others):

- it *reacts* to occurrences of events by issuing *actions* into the controlled environment,
- a *reaction* of the system has no duration: it is instantaneous,
- when no event occurs the system remains idle.

We will use in this article a simple version of GRAFCET with no forcing orders, macro-steps nor actions associated with the steps. GRAFCET is a graphical formalism suitable for specifying the control part of a real-time application: the word system means hereafter this control part.

2.1. Steps and Flows

A real-time application is made up of a number of activities which can be started or stopped: these activities are modelled by *steps* in GRAFCET. A step can be in two different states: *active* or *idle*. The scheduling of the steps depends on *boolean state variables* called *flows*. The evolution of the controlled environment is sensed through the changes of the values of the flows: the *edges*.

Let $\mathcal{E}$ be the set of steps of a particular grafcet, then the set of *active* steps at any time t is defined by the characteristic mapping

$$\Sigma \in 2^{\mathcal{E}} \tag{1}$$

Similarly, if $\mathcal{F}$ is the set of *flows*, the values of the flows at any time t are given by

$$\Phi \in 2^{\mathcal{F}} \tag{2}$$

Definition 1 *A* state s *is a pair* (Σ, Φ) *in* $2^{\mathcal{E}} \times 2^{\mathcal{F}}$.

2.2. Reaction of the system

The system that controls the application specified with the GRAFCET language *reacts* to edges (of the flows) by starting or stopping some steps. For any set A let

$$\overrightarrow{A} = \{\uparrow a, \downarrow a \mid a \in A\} \text{ and } \widetilde{A} = A \cup \overrightarrow{A}$$

The system reacts to events of $\overrightarrow{\mathcal{F}}$[1]. We denote $\uparrow e$ (resp. $\downarrow e$) the starting (resp. stopping) action for a step $e \in \mathcal{E}$. Then the starting and stopping action upon a reaction of the system is an element of $\wp(\overrightarrow{\mathcal{E}})$ (the set ot subsets of $\overrightarrow{\mathcal{E}}$).

[1] this is true if we consider that two events cannot occur simultaneously. Considering events to be in $\wp(\overrightarrow{\mathcal{F}})$ is another possible choice for which the semantics given below is always consistent.

The key issue is here to determine the next state: when we use the term *output actions* we mean the activations and deactivations of the steps (the usual "outputs" of a grafcet appear at another level). A reaction of the system to an event $\mathcal{I} \in \overrightarrow{\mathcal{F}}$ from state q brings about a set of output actions $\mathcal{O} \in \wp(\overrightarrow{\mathcal{E}})$ and leads to a new state q', which is formally written as

$$q \xrightarrow[\mathcal{O}]{\mathcal{I}}_{*} q' \tag{3}$$

We use the subscript $*$ under the right arrow to mean that this is a reaction (this will be opposed to *basic evolutions* in the sequel).

2.3. Specifying a system with the GRAFCET *language*

A GRAFCET specification of a real-time application defines (in a graphical way) what are the actions to be activated and stopped when the values of the flows change.

Definition 2 *A* grafcet γ *is a directed graph*

$$(\wp(\mathcal{E}), \wp(\mathcal{E}) \times R \times \wp(\mathcal{E}), \mathit{input}, \mathit{output})$$

where the vertices are sets of steps and the edges belong $\wp(\mathcal{E}) \times R \times \wp(\mathcal{E})$ *with* R *the set of transition conditions*[2]*. The mappings input and output are defined from* $\wp(\mathcal{E}) \times R \times \wp(\mathcal{E})$ *to* $\wp(\mathcal{E})$ *by* $\mathit{input}(E_1, r, E_2) = E_1$ *and* $\mathit{output}(E_1, r, E_2) = E_2$ *(they denote the input and output steps of a transition).*

Remark 1 The canonical representation of a grafcet as a set of transitions is the unique one which has as many transitions as the number of edges of the graphical representation.

In the sequel a grafcet is given by the set of transitions. Graphical examples with their corresponding sets of transitions are pictured on Fig. 1.

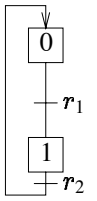
2.3.1. Transition conditions

We first introduce *transition conditions* defined by terms over the flows and their edges. The set of transition conditions R is the set of terms[3] over $\widetilde{\mathcal{F}} \cup \{\neg, \wedge, \vee, \mathit{tt}, \mathit{ff}, (,)\}$ inductively defined by:

- $\mathit{tt}, \mathit{ff} \in R$,
- $f \in R,\ \forall f \in \widetilde{\mathcal{F}}$,
- $\neg r, (r) \in R,\ \forall r \in R$ (we note $\bar{r}$ for $\neg r$),
- $r_1 \vee r_2, r_1 \wedge r_2 \in R, \forall r_1, r_2 \in R$.

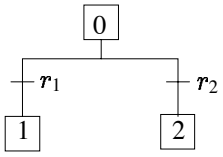
[2] this will be precisely defined later.

[3] we assume a particular grafcet under study.



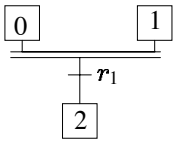
$\gamma_1 = (\{0\}, r_1, \{1\})$
$(\{1\}, r_2, \{0\})$

(a) Sequence



$\gamma_2 = (\{0\}, r_1, \{1\})$
$(\{0\}, r_2, \{2\})$

(b) OR (choice)



$\gamma_3 = (\{0,1\}, r_1, \{2\})$

(c) AND (parallelism)

Figure 1. *Examples of grafcets*

With each transition condition r we associate a value $I_{\Phi,\mathcal{I}}(r)$ in the boolean set $\{tt, ff\}$ depending upon $\Phi \in 2^{\mathcal{F}}$ and $\mathcal{I} \in \overrightarrow{\mathcal{F}}$. The interpretation $I_{\Phi,\mathcal{I}}(r)$ for $r \in R$ is defined inductively:

- $I_{\Phi,\mathcal{I}}(tt) = tt$, $I_{\Phi,\mathcal{I}}(ff) = ff$,
- $r \in \overrightarrow{\mathcal{F}}$, $I_{\Phi,\mathcal{I}}(r) = tt$ if $r \in \mathcal{I}$, ff otherwise,
- $r \in \mathcal{F}$, $I_{\Phi,\mathcal{I}}(r) = I_{\Phi,\mathcal{I}}\big((r)\big) = \big(\Phi(r) = tt\big)$,
- $I_{\Phi,\mathcal{I}}(\bar{r}) = \overline{I_{\Phi,\mathcal{I}}(r)}$,
- $I_{\Phi,\mathcal{I}}(r_1 \vee r_2) = I_{\Phi,\mathcal{I}}(r_1) \vee I_{\Phi,\mathcal{I}}(r_2)$,
- $I_{\Phi,\mathcal{I}}(r_1 \wedge r_2) = I_{\Phi,\mathcal{I}}(r_1) \wedge I_{\Phi,\mathcal{I}}(r_2)$.

2.3.2. Evolution rules

State changes are determined in GRAFCET by five *evolution rules* [COM 88] [DA 89]:

rule 1 the *initial situation* $\Sigma_0 \in 2^{\mathcal{E}}$ defines the active steps at the first instant,

rule 2 a transition is *enabled* if all its input steps are active; it is *clearable* if the two following conditions hold:

1. it is enabled,

2. the transition condition evaluates to true.

A *clearable* transition is immediatly cleared,

rule 3 clearing a transition yields the following changes:

- the input steps are stopped,
- the output steps are activated,

rule 4 if many transitions are simultaneously clearable, they are simultaneously cleared,

rule 5 if during an evolution a step must be both activated and stopped it remains activate (this is stronger than the classical rule which states this only for active steps).

Definition 3 *A* basic evolution *is a state change obtained by applying the five evolution rules simultaneously to each transition. We denote a basic evolution from state* q *on an occurrence of event* $\mathcal{I}$

$$q \xrightarrow[\mathcal{O}]{\mathcal{I}} q'$$

We do not use any subscript here to remind that this is a single evolution step.

2.4. Example 1

An example of a GRAFCET specification is given on Fig. 1.(a). with transition conditions $r_1 = a \vee \uparrow b$ and $r_2 = \downarrow a \vee \bar{b}$. The initial state of the system is $s_0 = (\Sigma_0, \Phi_0)$ with $\Sigma_0(0) = tt, \Sigma_0(1) = ff$, $\Phi_0(a) = \Phi_0(b) = ff$. In the remainding of the paper we will use the shorter notation $s_0 = (\{0\}, \bar{a}\bar{b})$ (i.e. Σ is replaced by $\Sigma^{-1}(tt)$ and for $a \in \mathcal{F}$ we use a if $\Phi(a) = tt, \bar{a}$ otherwise).

A possible interpretation of GRAFCET is the *without stability search* (WOSS) interpretation. It consists in processing new occurrences of events immediatly after each basic evolution. Then from state s_0 if event $\uparrow a$ occurs the basic evolution of the system is

$$s_0 \xrightarrow[\emptyset]{\uparrow a} \underbrace{(\{0\}, a\bar{b})}_{s_1} \tag{4}$$

Remark 2 In frequently used interpretations of GRAFCET the value of the transition condition r_1 would be true. In our interpretation, the value of this transition condition is $I_{\{\bar{a}\},\{\uparrow a\}}(a)$ which evaluates to false. However it is always possible to change this interpretation and use I' instead of I with $I'_{\Phi,\mathcal{I}} = I_{\Phi\oplus\mathcal{I},\mathcal{I}}$ where $\Phi \oplus \mathcal{I}$ denotes the updated values of the flows after the edges of $\mathcal{I}$ have occurred. The common interpretation assumes right continuity of the piecewise functions giving the values

of the flows. In this case a transition condition like $a \vee \uparrow a$ is equivalent to a in the sense that they are true at the very same time. Our choice is left continuity of the piecewise boolean functions: we consider that the values of the boolean variables will have changed only at the next state and that they are updated during the reaction. Thus $\uparrow a \vee a$ is no longer equivalent to a. The term a in a time condition refers to the last persistent value of the flow a. As our interpretation differs from the standard ones, we will use the term ^{R}GRAFCET when we consider it. The interested reader is refered to [DLR 91] for a complete discussion of this subject.

From state s_1, if no new event has occurred the transition condition r_1 is now true. s_1 is then a *non stable* state (since we can proceed further basic evolutions) and the evolution rules for the WOSS interpretation implies that clearable transitions be immediatly cleared[4]:

$$s_1 \xrightarrow[\{\uparrow 1, \downarrow 0\}]{\tau} \underbrace{(\{1\}, a\bar{b})}_{s_2} \tag{5}$$

On the contrary if event $\uparrow b$ has occurred during the basic evolution (4), the system reaches state s_3 instead of state s_2:

$$s_1 \xrightarrow[\{\uparrow 1, \downarrow 0\}]{\uparrow b} \underbrace{(\{1\}, ab)}_{s_3}$$

Now considering the reactive assumptions stated at the beginning of this section we have: 1) a reaction takes no time: thus no changes can occur in a null duration; 2) if no changes can occur during a basic evolution the system state changes with the silent move (5) without any new occurrences of events. This is not compatible with the reactive notions and thus a basic evolution in the WOSR interpretation can not be taken as a reaction.

The intuitive semantics we are going to present in section 3 implements another interpretation of GRAFCET: the *with stability search* (WSS) interpretation. With this interpretation of GRAFCET new events are not taken into account before a *stable* state has been reached. Then, from state s_0, the basic evolutions leading to a stable state are

$$s_0 \xrightarrow[\emptyset]{\uparrow a} s_1 \xrightarrow[\{\uparrow 1, \downarrow 2\}]{\tau} s_2$$

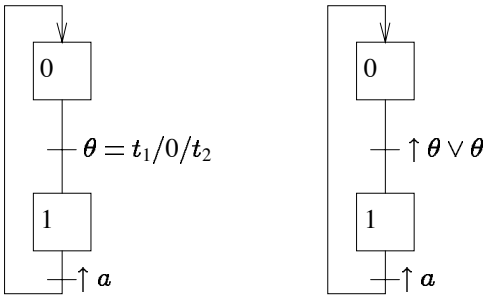
and the *reaction* of the system to event $\uparrow a$ is

$$s_0 \xrightarrow[\{\downarrow 0, \uparrow 1\}]{\uparrow a}{}^{*}\, s_2$$

[4] τ denotes the absence of occurrences of events ; such an evolution is unobservable.

2.5. Example 2

GRAFCET provides for *time condition* that can be used to delay actions and handle time units. They allow one to specify the application with a metric aspect of time that refers to the instants a step was started or stopped. A time condition is part of a transition condition: the grafcet pictured on Fig. 2.(a) has a time condition θ linking step 0 to step 1. A time condition θ is written $t_1/e/t_2$ with $t_1, t_2 \in \mathbb{R}^+ \cup \{\infty\}$ and $e \in \mathcal{E}$ (in standard GRAFCET $t_1, t_2 \in \mathbb{N} \cup \{\infty\}$). For $\theta = t_1/e/t_2$, we note $h(\theta) = e$ ($h(\theta)$ is the clock step of the time condition), $inf(\theta) = t_1$ and $sup(\theta) = t_2$.



(a) (b)

Figure 2. *A grafcet with time condition*

The intuitive semantics associated with a time condition is described on the time diagram Fig. 3:

1. at the initial instant θ is false,
2. step 0 is the only active step at the initial instant,
3. θ becomes true t_1 time units after step 0 has been activated[5]; at this date the transition condition involving θ is cleared and step 1 is started, step 0 is stopped,

[5] i.e., a stopwatch is started at the time step 0 is activated and after t_1 time units it ticks, step 0 being always active or not.

4. θ becomes false t_2 time units after step 0 has been stopped[6],

5. an occurrence of $\uparrow a$ starts step 0 again,

6. if step 0 is started again before t_2 time units have elapsed, θ is reset to false.

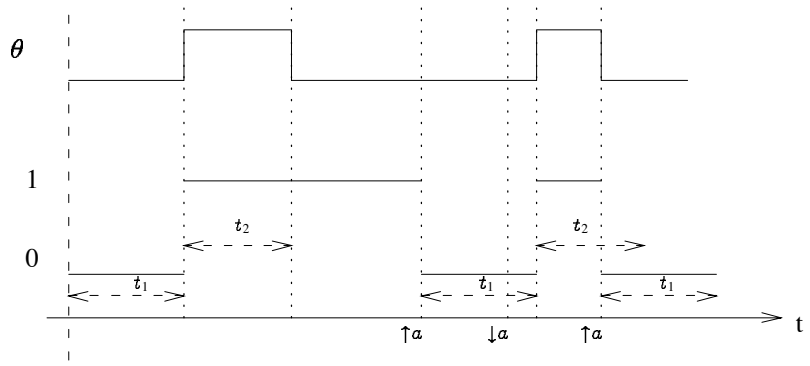


Figure 3. *Time diagram*

It is obvious that time condition are flows (boolean values). In ^{R}GRAFCET, time conditions are modelled with *time condition boolean variables*: the instants when the edges of these variables occur depend on the time elapsed since the starting or stopping of clock steps $h(\theta)$. In ^{R}GRAFCET, a time condition $\theta = t_1/e/t_2$ is semantically defined by a transition condition $\uparrow \theta \vee \theta$: the grafcet of Fig. 2.(a) is interpreted as Fig. 2.(b).

For any grafcet γ, let $\mathcal{T}$ be the set of all the time conditions. The values of the time conditions at any instant t are given by

$$\Theta \in 2^{\mathcal{T}} \tag{6}$$

The set of transition conditions R of GRAFCET is extended in ^{R}GRAFCET to the time conditions: transition conditions in ^{R}GRAFCET are terms built as described in section 2.3.1 with $\widetilde{\mathcal{F}}$ replaced by $\widetilde{\mathcal{F}} \cup \widetilde{\mathcal{T}}$. However the use of the operator $\neg$ (not) on time condition is forbidden (this is not for semantic reasons: one may think of introducing a new feature in ^{R}GRAFCET which is the negative time condition. This amounts to considering time intervals [ALL 81] and two operators *in* and *out*: such a notion has already been defined in [RLG 94]).

[6]another stopwatch is started at the instant step 0 is stopped and after t_2 time units it ticks unless step 0 has been activated during this period.

The interpretation of the new terms is straighforwardly extended to time conditions and the new interpretation is noted $I_{\Phi,\Theta,\mathcal{I}}(r)$. Finally a state is now a 3-tuple (Σ, Φ, Θ) in $2^{\mathcal{E}} \times 2^{\mathcal{F}} \times 2^{\mathcal{T}}$. We use S for the set of states $2^{\mathcal{E}} \times 2^{\mathcal{F}} \times 2^{\mathcal{T}}$ in the sequel.

2.6. Synchronism and asynchronism

Time conditions are viewed as boolean variables (like flows) the changes of which depend on an external clock. This is partial view of the time conditions as the metric part is left aside. This will enable us to build an abstract behavioural model for ^{R}GRAFCET programs that will be "timed" later on. So far time conditions are particular flows.

Occurrences of events from $\overrightarrow{\mathcal{F}}$ cannot be simultaneous as in the standard GRAFCET interpretations. But two events of $\overrightarrow{\mathcal{T}}$ may occur at the same time; moreover they sometimes do occur at the same time as in $\theta_1 = t_1/e/t_2$, $\theta_2 = t_1/e/t_2'$ for which $\uparrow\theta_1$ and $\uparrow\theta_2$ are always simultaneous. The complete treatment of these situations is defined in [CAS 95]. Then edges of time conditions variables are elements of $\wp(\overrightarrow{\mathcal{T}})$.

Finally, events of $\wp(\overrightarrow{\mathcal{T}})$ and $\overrightarrow{\mathcal{F}}$ cannot be simultaneous: we think that time condition events are to be processed with highest priority as they can not be delayed. Moreover, once the semantic model has been timed, the notion of an external clock is useless and state changes brought about by time conditions will be silent. From now on, a reaction has the following form

$$q \xrightarrow[\mathcal{O}]{\mathcal{I}}_{*} q' \qquad \mathcal{I} \in \overrightarrow{\mathcal{F}} \cup \wp(\overrightarrow{\mathcal{T}}),\ \mathcal{O} \in \wp(\overrightarrow{\mathcal{E}}) \tag{7}$$

Remark 3 In ^{R}GRAFCET transition conditions may involve the activities of the steps as in GRAFCET. This is developped in [CAS 95].

2.7. Stability

On the example of Fig. 2 we see that from state $q = (\{1\}, \bar{a}, \bar{\theta})$ there is a basic evolution to state q'

$$q = (\{1\}, \bar{a}, \bar{\theta}) \xrightarrow[\{\uparrow 0, \downarrow 1\}]{\uparrow a} (\{0\}, \bar{a}, \bar{\theta}) = q'$$

Is q' a stable state in the sense that there are no possible basic evolutions from this state? In standard GRAFCET interpretations this state is stable only if $t_1 > 0$, otherwise it is non stable. With the WSR standard interpretation and $t_1 = 0$ the next move is

$$q' \xrightarrow[\{\uparrow 1, \downarrow 0\}]{\tau} (\{1\}, \bar{a}, \theta) = q''$$

and if $t_1 > 0$ q' is stable.

In ^{R}GRAFCET q' is a stable state: to leave this state we must wait for $\uparrow\theta$ to occur. If $t_1 = 0$ then this event will occur 0 time unit after q' has been entered.

We will distinghish two types of unstability:

- intrinsic unstability which does not depend on any value (like t_1) is called *structural unstability*,
- unstability that depends on time condition values (the previous case) is called *temporal unstability*.

Temporal unstability will not be treated with the semantic rules and will be a property of the timed model defined in section 4.

3. Dynamic semantics for ^{R}GRAFCET

This section formalizes the calculus of basic evolutions and reactions. The semantics is given with conditional rules in the style of Plotkin [PLO 81].

3.1. Notations

For a set $A \subseteq \overrightarrow{\mathcal{E}} \cup \overrightarrow{\mathcal{F}} \cup \overrightarrow{\mathcal{T}}$, we note

$$A_\uparrow = \{a, \uparrow a \in A\} \quad \text{and} \quad A_\downarrow = \{a, \downarrow a \in A\}$$

for a set $A \subseteq \mathcal{E} \cup \mathcal{F} \cup \mathcal{T}$

$$\uparrow A = \{\uparrow a, a \in A\} \text{ and } \downarrow A = \{\downarrow a, a \in A\}$$

for $E, E', E'' \subseteq \mathcal{E}$

$$\Delta(E, E', E'') = \Big(\uparrow (E' \setminus E) \cup \downarrow (E \setminus E')\Big) \setminus \uparrow (E'')$$

and for $\mathcal{O}', \mathcal{O}'' \subseteq \overrightarrow{\mathcal{E}}$

$$\mathcal{O}' \uplus \mathcal{O}'' = \Big\uparrow (\mathcal{O}_\uparrow) \cup \Big\downarrow \big(\mathcal{O}_\downarrow \setminus \mathcal{O}_\uparrow\big) \text{ where } \mathcal{O} = \mathcal{O}' \cup \mathcal{O}''$$

The previous definitions will be used to calculate the steps that are to be activated and stopped on a reaction or basic evolutions. $\Delta(E, E', E'')$ allows us to calculate the set of steps that must be activated and deactivated when firing a transition. If E and E' respectively denote the input and output steps of a transition and E'' the set of active steps in the current state of the system, $\Delta(E, E', E'')$ gives the set of output events corresponding to the firing of the transition: $\uparrow i$ (resp. $\downarrow i$) means that step i is to be activated (resp. deactivated). According to rule 5, the activated steps are those which were not active in the last state. The operator $\uplus$ extends Δ to simultaneous firing of transitions. If taken independently, the firing of two transitions entail output events $\mathcal{O}$ and $\mathcal{O}'$, then when fired simultaneously, rule 5 applies: a step is activated if is activated by at least one of the firing, and deactivated if is not simulatneously activated.

To update the boolean variables we need the operator $\oplus_f$ over $2^{\mathcal{F}} \times \overrightarrow{\mathcal{F}}$ with value in $2^{\mathcal{F}}$ defined by :

$$\begin{aligned}(\Phi \oplus_f \mathcal{I})(f) &= \Phi(f) \; if \; \uparrow f \notin \mathcal{I} \wedge \downarrow f \notin \mathcal{I} \\ (\Phi \oplus_f \mathcal{I})(f) &= \bar{f} \; if \; \Phi(f) \; and \; \downarrow f \in \mathcal{I} \\ (\Phi \oplus_f \mathcal{I})(f) &= f \; if \; \overline{\Phi(f)} \; and \; \uparrow f \in \mathcal{I}\end{aligned}$$

Similarly we introduce $\oplus_e$ and $\oplus_t$ over $2^{\mathcal{E}} \times \overrightarrow{\mathcal{E}}$ and $2^{\mathcal{T}} \times \overrightarrow{\mathcal{T}}$ with values in $2^{\mathcal{E}}$ and $2^{\mathcal{T}}$ to update the activities of the steps and the time condition variables. Finally for $q = (\Sigma, \Phi, \Theta) \in S$, $\mathcal{I} \in \wp(\overrightarrow{\mathcal{T}}) \cup \overrightarrow{\mathcal{F}}$ and $\mathcal{O} \in \wp(\overrightarrow{\mathcal{E}})$, we use the following abbreviation

$$q \oplus (\mathcal{O}, \mathcal{I}) = (\Sigma \oplus_e \mathcal{O}, \Phi \oplus_f \mathcal{I}, \Theta \oplus_t \mathcal{I})$$

3.2. Syntax of ^{R}GRAFCET

A *grafcet* γ in ^{R}GRAFCET is a directed graph the edges of which are elements of $\wp(\mathcal{E}) \times R \times \wp(\mathcal{E})$. As we want to give a structural operational semantics for ^{R}GRAFCET we define an abstract ambiguous grammar to describe the element of ^{R}GRAFCET (the axiom is Γ):

$$\Gamma ::= (E, r, E) \tag{8}$$

$$\mid \; \Gamma \,||\, \Gamma \tag{9}$$

with $E \in \wp(\mathcal{E}), r \in R$. If $g = (E_1, r, E_2) \in \gamma$, E_1 is the set of *input steps* and E_2 the *output steps*.

3.3. Semantics

The calculus of basic evolution is formally defined by the structural operational semantics (SOS) given in Fig. 4. These rules express the five evolution rules listed in section 2.3.2.

For a grafcet γ, a state q and an event $\mathcal{I}$, the basic evolution from q on the occurrence of $\mathcal{I}$ written

$$q \xrightarrow[\mathcal{O}]{\mathcal{I}} q'$$

is the only one[7] that satisfies

$$\gamma \vdash \; q \xrightarrow[\mathcal{O}]{\mathcal{I}} q'$$

[7] unicity and existence are discussed hereafter.

$$(f)\ \frac{E_1 \subseteq \Sigma^{-1}(tt) \wedge \mathcal{I}_{(\Phi,\Theta,\mathcal{I})}(r) = tt}{(E_1, r, E_2) \vdash\ (\Sigma,\Phi,\Theta) \xrightarrow[\Delta(E_1,E_2,\Sigma^{-1}(tt))]{\mathcal{I}} (\Sigma,\Phi,\Theta) \oplus (\Delta(E_1, E_2, \Sigma^{-1}(tt)), \mathcal{I})}$$

$$(p)\ \frac{E_1 \not\subseteq \Sigma^{-1}(tt) \vee \mathcal{I}_{(\Phi,\Theta,\mathcal{I})}(r) = ff}{(E_1, r, E_2) \vdash\ (\Sigma,\Phi,\Theta) \xrightarrow[\emptyset]{\mathcal{I}} (\Sigma,\Phi,\Theta) \oplus (\emptyset, \mathcal{I})}$$

$$(g)\ \frac{\gamma_1 \vdash\ q \xrightarrow[\mathcal{O}']{\mathcal{I}} q' \ \wedge\ \gamma_2 \vdash\ q \xrightarrow[\mathcal{O}'']{\mathcal{I}} q''}{\gamma_1 \parallel \gamma_2 \vdash\ q \xrightarrow[\mathcal{O}' \uplus \mathcal{O}'']{\mathcal{I}} q \oplus (\mathcal{O}' \uplus \mathcal{O}'', \mathcal{I})}$$

Figure 4. *Operational semantics for* ^{R}GRAFCET

3.4. Properties of the semantics

Theorem 1 *The operational semantics given for* ^{R}GRAFCET *is deterministic and complete. Moreover, the result* $\gamma \vdash\ q \xrightarrow[\mathcal{O}]{\mathcal{I}} q'$ *does not depend on the derivation tree*[8] *used for program* Γ.

Proof of Theorem 1

Completeness is obvious from the rules of Fig. 4. Determinism relies on a property of operator $\uplus$. In a previous version of this article [CAS 96] the claim that commutativity of the operator $\uplus$ was sufficient to ensure determinism was wrong: this mistake has been pointed out by Charles André. We give here a correct proof for determinism.

We first establish that $\uplus$ is associative. For this we need the following algebraic laws:

$$\begin{aligned}
\uparrow A \cup \uparrow B &= \uparrow (A \cup B) \text{ (same for } \downarrow) \\
A_{\uparrow} \cup B_{\uparrow} &= (A \cup B)_{\uparrow} \text{ (same for } \downarrow) \\
(\downarrow A)_{\uparrow} &= \emptyset \text{ (the converse holds)} \\
(\uparrow A)_{\uparrow} &= A \text{ (same for } \downarrow)
\end{aligned}$$

Now let $\mathcal{O}', \mathcal{O}'', \mathcal{O}'''$ be subsets of $\overrightarrow{\mathcal{E}}$.

$$(\mathcal{O}' \uplus \mathcal{O}'') \uplus \mathcal{O}''' = \uparrow \left(\left((\mathcal{O}' \uplus \mathcal{O}'') \cup \mathcal{O}''' \right)_{\uparrow} \right) \bigcup$$

[8] we remind the reader that the ^{R}GRAFCET grammar is ambiguous.

$$\Big\downarrow\Big(\Big((\mathcal{O}' \uplus \mathcal{O}'') \cup \mathcal{O}'''\Big)_{\downarrow} \setminus \Big((\mathcal{O}' \uplus \mathcal{O}'') \cup \mathcal{O}'''\Big)_{\uparrow}\Big)$$

From the previous laws we infer that

$$\begin{aligned}\Big((\mathcal{O}' \uplus \mathcal{O}'') \cup \mathcal{O}'''\Big)_{\uparrow} &= \Big(\mathcal{O}' \cup \mathcal{O}'' \cup \mathcal{O}'''\Big)_{\uparrow}\\ \Big((\mathcal{O}' \uplus \mathcal{O}'') \cup \mathcal{O}'''\Big)_{\downarrow} &= \Big(\Big(\mathcal{O}' \cup \mathcal{O}''\Big)_{\downarrow} \setminus \Big(\mathcal{O}' \cup \mathcal{O}''\Big)_{\uparrow}\Big) \cup \mathcal{O}'''_{\downarrow}\end{aligned}$$

which yields to

$$\begin{aligned}(\mathcal{O}' \uplus \mathcal{O}'') \uplus \mathcal{O}''' &= \Big\uparrow\Big(\Big(\mathcal{O}' \cup \mathcal{O}'' \cup \mathcal{O}'''\Big)_{\uparrow}\Big)\\ &\cup \Big\downarrow\Big(\Big(\Big(\mathcal{O}' \cup \mathcal{O}''\Big)_{\downarrow} \setminus \Big(\mathcal{O}' \cup \mathcal{O}''\Big)_{\uparrow}\Big) \cup \mathcal{O}'''_{\downarrow}\Big) \setminus \Big(\Big(\mathcal{O}' \cup \mathcal{O}'' \cup \mathcal{O}'''\Big)_{\uparrow}\Big)\end{aligned}$$

and as $(A \setminus B) \setminus (C \cup D) = \big(A \setminus (B \cup D)\big) \setminus (C \cup D)$ we obtain

$$\begin{aligned}(\mathcal{O}' \uplus \mathcal{O}'') \uplus \mathcal{O}''' &= \Big\uparrow\Big(\mathcal{O}' \cup \mathcal{O}'' \cup \mathcal{O}'''\Big)_{\uparrow}\\ &\cup \Big\downarrow\Big(\Big(\mathcal{O}' \cup \mathcal{O}'' \cup \mathcal{O}'''\Big)_{\downarrow} \setminus \Big(\mathcal{O}' \cup \mathcal{O}'' \cup \mathcal{O}'''\Big)_{\uparrow}\Big)\end{aligned}$$

A similar calculus can be carried out for $\mathcal{O}' \uplus (\mathcal{O}'' \uplus \mathcal{O}''')$ ending with the same result which proves that $\uplus$ is associative.

Now remark that in rule (g) of Fig. 4 the new state $q \oplus (\mathcal{O}' \uplus \mathcal{O}'', \mathcal{I})$ depends on the previous one q and on $\mathcal{O}' \uplus \mathcal{O}''$ and not on q' and q'' written in the premisses. This ensures determinism. The commutativity (obvious) of the operator $\uplus$ ensures that whatever the order the elements of the directed graph are listed in, the semantics of a grafcet is unique:

$$\gamma_1 \,||\, \gamma_2 \vdash t \Longleftrightarrow \gamma_2 \,||\, \gamma_1 \vdash t$$

□

3.5. Structural stability

We define *structural stability* for a state $q = (\Sigma, \Phi, \Theta)$, a grafcet γ and an output event $\mathcal{O} \in \wp(\overrightarrow{\mathcal{E}})$, denoted $\mathcal{K}_{(q,\mathcal{O})}(\gamma)$ inductively by :

- if $\gamma = (E_1, r, E_2)$ then

$$\mathcal{K}_{(q,\mathcal{O})}(\gamma) = E_1 \not\subseteq \Sigma^{-1}(tt) \vee \big(I_{\Phi,\Theta,\mathcal{O}}(r) = ff\big)$$

- if $\gamma = \gamma_1 \,||\, \gamma_2$ then

$$\mathcal{K}_{(q,\mathcal{O})}(\gamma) = \mathcal{K}_{(q,\mathcal{O})}(\gamma_1) \wedge \mathcal{K}_{(q,\mathcal{O})}(\gamma_2)$$

Definition 4 *A pair* $(q, \mathcal{O})$ *with* $\mathcal{O} \in \wp(\overrightarrow{\mathcal{E}})$ *is* stable *for a grafcet* γ *iff* $\mathcal{K}_{(q,\,\mathcal{O})}(\gamma) = tt$. *A pair* $(q, \mathcal{I})$ *with* $\mathcal{I} \in \overrightarrow{\mathcal{F}} \cup \wp(\overrightarrow{\mathcal{T}})$ *is non stable*.

3.6. Calculus of a reaction

Definition 5 *The*[9] reaction *from state* q_0 *on an occurrence of event* $\mathcal{I} \in \wp(\overrightarrow{\mathcal{T}}) \cup \overrightarrow{\mathcal{F}}$ *is the sequence* $t_0 t_1 t_2 \ldots t_n \ldots$ *of basic evolutions from* q_0 *such that* $t_0 = q_0 \xrightarrow[\mathcal{O}_1]{\mathcal{I}=\mathcal{O}_0} q_1$ *and for each* $i \geq 0$, $t_i = q_i \xrightarrow[\mathcal{O}_{i+1}]{\mathcal{O}_i} q_{i+1}$, *and* $\mathcal{K}_{(q_i,\,\mathcal{O}_i)}(\gamma) = ff$.

This sequence may be finite if there is a stable pair $(q_i, \mathcal{O}_i)$, and infinite otherwise. If the sequence is finite we write

$$q_0 = q \xrightarrow[\mathcal{O}_1]{\mathcal{I}} q_1 \xrightarrow[\mathcal{O}_2]{\mathcal{O}_1} \cdots \xrightarrow[\mathcal{O}_n]{\mathcal{O}_{n-1}} q_n \xrightarrow[\mathcal{O}_{n+1}]{\mathcal{O}_n} q_{n+1}$$

with $\mathcal{K}_{\left(q_{n+1},\,\mathcal{O}_{n+1}\right)}(\gamma)$. Otherwise

$$q_0 = q \xrightarrow[\mathcal{O}_1]{\mathcal{I}} q_1 \xrightarrow[\mathcal{O}_2]{\mathcal{O}_1} \cdots \xrightarrow[\mathcal{O}_n]{\mathcal{O}_{n-1}} q_n \xrightarrow[\mathcal{O}_{n+1}]{\mathcal{O}_n} \cdots$$

and $\forall i \in \mathbb{N}, \mathcal{K}_{(q_i,\,\mathcal{O}_i)}(\gamma) = ff$. We denote I the set of indices of the state of a reaction: if it is finite then $I = [0, n], n \geq 1$, otherwise $I = \mathbb{N}$.

Definition 6 *The* orbit $(q, \mathcal{I})^{\nearrow}$ *of a reaction from* q *on* $\mathcal{I}$ *is the set* $\{(q_n, \mathcal{O}_n), n \in I\}$.

Proposition 1 $\forall (q, \mathcal{I}) \in S \times \wp(\overrightarrow{\mathcal{T}}) \cup \overrightarrow{\mathcal{F}}$, $(q, \mathcal{I})^{\nearrow}$ *contains at most one stable pair*

$$\forall \sigma, \sigma' \in (q, \mathcal{I})^{\nearrow}, \qquad \mathcal{K}_\gamma(\sigma) \wedge \mathcal{K}_\gamma(\sigma') \Longrightarrow \sigma = \sigma'$$

and if there is a stable pair in $(q, \mathcal{I})^{\nearrow}$, *this set is finite*.

Proof of proposition 1
Direct from the definition 5 of a reaction. □

Now let $\overrightarrow{E} = \wp(\overrightarrow{\mathcal{T}}) \cup \overrightarrow{\mathcal{F}} \cup \wp(\overrightarrow{\mathcal{E}})$. To calculate the orbit we introduce a function $\mathcal{N}_\gamma$ for a given grafcet γ of ^{R}GRAFCET defined by :

$$\begin{array}{rcccl} \mathcal{N}_\gamma & : & S \times \overrightarrow{E} & \longrightarrow & S \times \overrightarrow{E} \\ & & (q, e) & \longmapsto & (q', \mathcal{O}) \text{ such that } q \xrightarrow[\mathcal{O}]{e} q' \end{array}$$

[9] determinism and completeness of the semantics ensure existence and uniqueness of the reaction.

Remark 4 $\mathcal{N}_\gamma$ is a total function as the operational semantics is deterministic and complete. $\mathcal{O}$ is an output event from $\wp(\overrightarrow{\mathcal{E}})$.

We extend $\mathcal{N}_\gamma$ to $\wp(S \times \overrightarrow{E})$:

$$\begin{array}{rcccc} \widetilde{\mathcal{N}_\gamma} & : & \wp(S \times \overrightarrow{E}) & \longrightarrow & \wp(S \times \overrightarrow{E}) \\ & & X & \longmapsto & \{\mathcal{N}_\gamma(x), x \in X\} \end{array}$$

Then, in the ordered set $\langle \wp(S \times \overrightarrow{E}), \subseteq \rangle$, the function $\widetilde{\mathcal{N}_\gamma}$ is monotonic. From the well-known Tarski's theorem [TAR 55] we deduce that the equation $X = \varphi(q, \mathcal{I})(X)$ with

$$\varphi(q, \mathcal{I})(X) = (q, \mathcal{I}) \cup \widetilde{\mathcal{N}_\gamma}(X)$$

has a least fixed point $\mu.\varphi(q, \mathcal{I})$. As $\wp(S \times \overrightarrow{E})$ is finite with minimum element $\emptyset$ there exists $k \in \mathbb{N}$ such that $\mu.\varphi(q, \mathcal{I}) = \varphi(q, \mathcal{I})^k(\emptyset)$. Moreover the following theorem holds :

Theorem 2 $\forall \sigma \in S \times \wp(\overrightarrow{\mathcal{T}}) \cup \overrightarrow{\mathcal{F}}, \quad \sigma^\nearrow = \mu.\varphi(\sigma)$

Proof of theorem 2
Let $L_0 = f_\sigma(\emptyset)$ and $\forall i \geq 0,\ L_{i+1} = \varphi(\sigma)(L_i)$ then $\mu.\varphi(\sigma) = \cup_{i \in \mathbb{N}}(L_i)$. It suffices to prove:

1. $\forall (q, \mathcal{I}) \in \sigma^\nearrow, \exists n \in \mathbb{N}, (q, \mathcal{I}) \in L_n$,
2. $\forall n \in \mathbb{N}$ si $(q, \mathcal{I}) \in L_n$ alors $(q, \mathcal{I}) \in \sigma^\nearrow$.

This is straightforward by induction on $\mathbb{N}$. □

3.7. Reaction of the system

A reaction of the system is a sequence of basic evolutions. For $q \in S$ and $\mathcal{I} \in \overrightarrow{\mathcal{F}} \cup \wp(\overrightarrow{\mathcal{T}})$, the reaction of the system to $\mathcal{I}$ is :

$$q \xrightarrow[\mathcal{O}]{\mathcal{I}}_* q'$$

where:

- if $q^\nearrow$ is infinite, $q' = q_\perp$ and $\mathcal{O} = \mathcal{O}_\perp$ (the undefined state and undefined output event),

- otherwise $q^{\nearrow}$ is finite, the reaction $\rho = e_0 e_1 e_2 \ldots e_n$ converges ; let $q_{n+1} = (\Sigma_{n+1}, \Phi_{n+1}, \Theta_{n+1})$, the steps to be activated and stopped and the updated values of the boolean time condition variables are defined using the initial and final states q and q_{n+1}. The reaction is then defined to be $q \xrightarrow[\mathcal{O}]{\mathcal{I}}_* q'$ where $q' = (\Sigma_{n+1}, \Phi_{n+1}, \Theta')$ and:

$$\mathcal{O} = \underbrace{\uparrow\left(\Sigma_{n+1}^{-1}(tt) \setminus \Sigma^{-1}(tt)\right)}_{\text{(started steps)}} \bigcup \underbrace{\downarrow\left(\Sigma^{-1}(tt) \setminus \Sigma_{n+1}^{-1}(tt)\right)}_{\text{(stopped steps)}}$$

and $\Theta'(\theta) = ff$ if $h(\theta) \in \mathcal{O}_{\uparrow}$, otherwise $\Theta'(\theta) = \Theta_{n+1}(\theta)$.

This way, a step is considered to be activated (resp. stopped) if it was idle (resp. active) in q and now is active (resp. idle) in q'. Similarly, we consider time condition to be set according to external time: the correction to the final state is defined according to what happened on the reaction and the values of the boolean time condition variables are set accordingly.

4. Finite state modeling of ^{R}GRAFCET programs

The semantic model for ^{R}GRAFCET program is given only for programs with at most one time condition (general case is in [CAS 95]). For a time condition θ we introduce two clock variables

- x_θ measures the elapsed time since the last start action on $h(\theta)$,
- y_θ, measures the elapsed time since the last stop action on $h(\theta)$.

4.1. Awaited events

It is clear that a way of defining a semantic model for ^{R}GRAFCET programs would consist in exploring from any state all the possible transitions brought about by all the events of $\wp(\overrightarrow{T}) \cup \overrightarrow{\mathcal{F}}$. In fact, not all events are possible from any state and it reduces the size of the model if we care for not introducing impossible transitions.

From a state $q = (\Sigma, \Phi, \Theta)$, we define the *awaited events* to be

$$\mathcal{I}(q) = \downarrow\left(\Phi^{-1}(tt)\right) \cup \uparrow\left(\Phi^{-1}(ff)\right) \cup T$$

where $T = \{\uparrow\theta\}$ if $h(\theta) \in \Sigma^{-1}(tt) \wedge \theta \in \Theta^{-1}(ff)$, $T = \{\downarrow\theta\}$ if $h(\theta) \in \Sigma^{-1}(ff) \wedge \theta \in \Theta^{-1}(tt)$ and $T = \emptyset$ otherwise.

Example For the grafcet of Fig. 2.(b), $\mathcal{I}(s_1) = \{\uparrow\theta, \uparrow a\}$. For state $s_5 = (\{1\}, \bar{a}, \bar{\theta})$, $\mathcal{I}(s_5) = \{\uparrow a\}$.

4.2. Liveness

The system can stay in a state as long as no edges occur. Edges for time conditions θ occur at definite date when $x_\theta = inf(\theta)$ and $y_\theta = sup(\theta)$. Then staying in state $q = (\Sigma, \Phi, \Theta)$ entails the following inequations to hold:

- $x_\theta \leq inf(\theta)$ if $\theta \in \Theta^{-1}(ff) \wedge h(\theta) \in \Sigma^{-1}(tt)$,
- $y_\theta \leq sup(\theta)$ if $\theta \in \Theta^{-1}(tt) \wedge h(\theta) \in \Sigma^{-1}(ff)$,
- tt otherwise.

We note $max^+(q)$ the inequation associated with state q. Similarly, we need max^- where $\leq$ is replaced by $<$.

4.3. Behavioural model: Timed automaton

The behavioural model for an ^{R}GRAFCET program γ is a *timed automaton* [NSY 92] [AD 90]. Let $\mathcal{E}$, $\mathcal{F}$ et $\mathcal{T}$ be the sets of steps, boolean variables and time condition variables for a grafcet γ. The initial state of the system is defined by $E_0 \subseteq \mathcal{E}$. The behavioural model for grafcet γ is the timed automaton $\mathcal{A}_\gamma = \langle S, X, s_0, act, \rightarrow \rangle$ where:

- $S = 2^{\mathcal{E}} \times 2^{\mathcal{F}} \times 2^{\mathcal{T}} \cup \{q_\perp\}$ is the set of *states*,
- $s_0 = (\Sigma_0, \Phi_0, \Theta_0) \in S$ is defined by
  - $\Sigma_0^{-1}(tt) = E_0, \Sigma_0^{-1}(ff) = \mathcal{E} \setminus E_0$
  - $\Phi_0(f) = ff, \forall f \in \mathcal{F}$,
  - $\Theta_0(\theta) = ff, \forall \theta \in \mathcal{T}$,
- $X = \{x_\theta, y_\theta,\ \theta \in \mathcal{T}\}$ is the set of *timers* of the automaton,
- $act \in S \rightarrow \phi$ is such that $\forall q = (\Sigma, \Phi, \Theta) \in S, act(q) = max^+(q)$ (ϕ is the set of boolean terms [NSY 92] over the variables of X),
- $\rightarrow \in S \times (\phi \times A \times \wp(X)) \times S$ where $A = \overrightarrow{\mathcal{F}} \cup \{tick\}$ is defined inductively as follows:
  1. $$\frac{a \in \mathcal{I}(q) \cap \overrightarrow{\mathcal{F}},\quad q \xrightarrow[\mathcal{O}]{a}{}^* q'}{q \xrightarrow{max^-(q), a, A_X \cup D_X} q'}$$

2. $a \in \mathcal{I}(q) \cap \overrightarrow{T}$

$$\frac{q \xrightarrow[\mathcal{O}]{a}{}^{*} q'}{q \xrightarrow{\eta(a),tick,A_X \cup D_X} q'}$$

where $A_X = \{x_\theta, \theta \in \mathcal{O}_\uparrow\}$ and $D_X = \{y_\theta, \theta \in \mathcal{O}_\downarrow\}$ et $\eta(\uparrow \theta) = (x_\theta = inf(\theta))$ and $\eta(\downarrow \theta) = (y_\theta = sup(\theta))$.

For the grafcet of Fig. 2.(b) we obtain the timed automaton given Fig. 5 [10]. The states information is listed in Table 1.

state	state information	*act*
s_0	$(\{0\}, \bar{a}, \bar{\theta})$	$x_\theta \leq t_1$
s_1	$(\{1\}, \bar{a}, \theta)$	$y_\theta \leq t_2$
s_2	$(\{0\}, a, \bar{\theta})$	$x_\theta \leq t_1$
s_3	$(\{1\}, a, \theta)$	$y_\theta \leq t_2$
s_4	$(\{1\}, a, \bar{\theta})$	tt
s_5	$(\{1\}, \bar{a}, \bar{\theta})$	tt

Table 1. *States information*

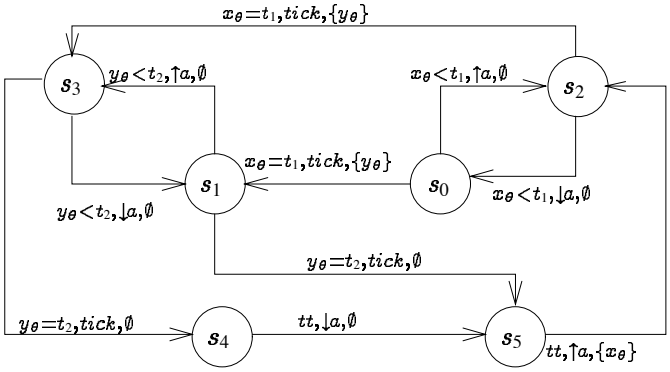


Figure 5. *Timed automaton*

[10] the automaton given in [CAS 96] contains few mistakes that are corrected here.

Remark 5 If $t_1 = 0$ (Cf. 2.7), we move from state s_5 to state s_0 on an occurrence of $\uparrow a$: in this state the condition $act(s_0) = x_\theta \leq t_1$ is true if time does not pass: moving to state s_1 takes no time.

4.4. Stability

Structural stability It is quite obvious that a couple $(q, \mathcal{I})$ is non stable if $(q, \mathcal{I})^{\nearrow}$ is infinite. In this case the corresponding reaction is defined as in section 3.7.

Let γ be a particular grafcet. An endless (undefined) reaction $q \xrightarrow[\mathcal{O}_\perp]{\mathcal{I}}_* q_\perp$ is possible only if state $q_\perp$ is *reachable* in the associated automaton $\mathcal{A}_\gamma$.

We will not discuss here the decidability of this problem [ACH$^+$ 95] [KPSY 93] [RR 96b] [RR 96a]: in fact it has so far been proved decidable only for bounded reachability (i.e. reachability within a given time interval).

Temporal stability Temporal stability occurs only if there is a cycle in the automaton $\mathcal{A}_\gamma$ on which time does not progress. Checking the automaton for temporal stability then amounts to a reachability problem within a time interval of null duration. Again we refer the reader to [ACH$^+$ 95] [KPSY 93] [RR 96b] [RR 96a] for a discussion on these problems.

5. Conclusion

The work presented in this paper differs from the ones where GRAFCET is translated in a reactive language [RR 94] [AP 92] [ML 92] since we model the metric aspect of time. The semantics given above is based on the idea of [AG 94] where the evolution rules are characterized by fixpoints[11]; our semantics include the time conditions.

We have shown in this article that an operational semantics could be defined for GRAFCET giving this language a formal background. This semantics associates with a grafcet a timed automaton. A compiler from ^{R}GRAFCET to timed automata has been written in CAML [LW 93] [WL 93]. On non trivial examples the size of the associated automaton is rather small with respect to the maximum number of possible states ($2^{|\mathcal{E}|+|\mathcal{F}|+|\mathcal{T}|}$) as shown on Table 2. We may then think of verifying "real" grafcets with tools like VALET [RUS 96] or KRONOS [YOV 93] (although those tools can right now only handle small systems).

Moreover, the model (namely a timed automaton) can be enriched to include the actions of GRAFCET to obtain a more accurate model of the system: the enriched model is then a hybrid system [ACH$^+$ 95] [NOSY 93]. We then use the tool VALET [RUS 96] developped at L.A.N. (Nantes) to check properties on particular classes of hybrid

[11] the semantics of reactive languages is also defined this way.

Steps	Flows	Time conditions	Edges	$2^{\lvert\mathcal{E}\rvert+\lvert\mathcal{F}\rvert+\lvert\mathcal{T}\rvert}$	Number of states
3	1	2	3	64	15
5	2	3	6	1024	142
8	3	0	7	2048	40
12	7	1	11	1 048 576	1952

Table 2. *Sizes of the automata*

systems [RR 96b] [RR 96a]: this tool provides for the use of stopwatches (i.e. clocks that can be stopped and restarted later) when KRONOS only handles timed automata.

The author wishes to thank Charles André for his careful reading of the many versions of the paper and the other anonymous referees for their comments which helped us to improve many parts of this work.

6. References

[AA 92] ADEPA-AFCET. *Le Grafcet*. Cépadues, 1992.

[ACH+ 95] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science B*, 137, January 1995.

[AD 90] Rajeev Alur and David Dill. Automata for modelling real-time systems. In *Lecture Notes in Computer Science*, volume 443, pages 322–336. Springer–Verlag, July 1990.

[AFC 77] AFCET. Normalisation de la représentation du cahier des charges d'un automatisme logique. *Automatique et Informatique Industrielle*, 1977.

[AFN 82] AFNOR. Norme C03–190, 1982. Paris (FRANCE).

[AG 94] Charles André and Daniel Gaffé. Événements et Conditions en GRAFCET. *Automatique Productique et Informatique Industrielle*, 28(4):331–352, 1994.

[ALL 81] J.F. Allen. An interval-based representation of temporal knowledge. In *Proceedings of the IJCAI'81*, pages 221–226, Vancouver, August 1981.

[AP 92] Charles André and Marie-Agnès Peraldi. GRAFCET et langages synchrones. In *GRAFCET'92 Conf.*, Paris, march 1992.

[BB 91] Albert Benveniste and Gérard Berry. The synchronous approach to reactive and real-time systems. *Proceedings of the IEEE*, 79(9):1270–1282, september 1991.

[BD 91] Frédéric Boussinot and Robert De Simone. The ESTEREL language. *Proceedings of the IEEE*, 79(9):1293–1304, september 1991.

[BG 91] Gérard Berry and Georges Gonthier. The ESTEREL synchronous language: design, semantics, and implementation. *Science of Computer Programming*, 1991.

[BRA 83] G. W. Brams. *Réseaux de Pétri: Théorie et Pratique*. MASSON, 1983. Tomes 1–2.

[CAS 95] Franck Cassez. ^{R}GRAFCET : Un GRAFCET réactif. Research Report RR –95–01, Département d'Informatique, Université de Bretagne Occidentale, 6, Avenue Le Gorgeu B.P. 809 29285 Brest Cedex (FRANCE), 1995.

[CAS 96] Franck Cassez. Une sémantique pour GRAFCET réactif. In *Modélisation des systèmes réactifs*, pages 373–380, Brest (FRANCE), March 1996.

[COM 88] International Electrotechnical Commission. Preparation of function charts for control systems, IEC 848, 1988.

[COM 93] International Electrotechnical Commission. IEC 1131 standard for programmable controllers, part 3: Programming languages, 1993.

[CPHP 87] Paul Caspi, Daniel Pilaud, Nicolas Halbwachs, and John A. Plaice. LUSTRE: a declarative language for programming synchronous systems. In *Proceedings of the 14^{th} ACM Symposium on Principles of Programming Languages*, Munich (GERMANY), january 1987.

[CR 95] Franck Cassez and Olivier Roux. Compilation of the ELECTRE reactive language into finite transition systems. *Theoretical Computer Science B*, 146(1–2):109–143, July 1995.

[DA 89] René David and Hassane Alla. *Du* GRAFCET *aux réseaux de Petri*. Editions HERMES, Paris, 1989.

[DLR 91] B. Denis, J-J. Lesage and J-M. Roussel. A boolean algebra for a formal expression of events in logical systems. *Proceedings of the IMACS Mathmod*, Vienna (Austria), pages 859–862, February 1994.

[ER 85] Jean-Pierre Elloy and Olivier Roux. ELECTRE: A language for control structuring in real time. *The Computer Journal*, 28(5), july 1985.

[HAR 87] David Harel. STATECHARTS*: a Visual Formalism for Complex Systems*, volume 8, pages 231–274. North Holland, june 1987.

[HCRP 91] Nicolas Halbwachs, Paul Caspi, Pascal Raymond, and Daniel Pilaud. The synchronous dataflow language LUSTRE. *Proceedings of the IEEE*, 79(9):1304–1320, september 1991.

[KPSY 93] Y. Kesten, A. Pnueli, J. Sifakis, and S. Yovine. Integration graphs: a class of decidable hybrid systems. In R.L. Grossman, Nerode, A.P. Ravn, and H. Richel, editors, *Workshop on Theory of Hybrid Systems*, pages 179–208, Lyngby, Denmark, June 1993. Lecture Notes in Computer Science 736, Springer-Verlag.

[LBBG 86] Paul Le Guernic, Albert Benveniste, Patricia Bournai, and Thierry Gautier. SIGNAL: a data-flow oriented language for signal processing. *IEEE transactions on ASSP*, ASSP-34(2):362–374, 1986.

[LLGL 91] Paul Le Guernic, Michel Le Borgne, Thierry Gautier, and Claude Le Maire. Programming real-time applications with SIGNAL. *Proceedings of the IEEE*, 79(9):1321–1336, september 1991.

[LW 93] Xavier Leroy and Pierre Weis. *Manuel de référence du langage* CAML. InterÉditions, 1993.